### ****ML Application: XGBoost Training – Pandas vs. Dask****

#### ****Model Type:****

* XGBoost (Extreme Gradient Boosting) for supervised learning (classification).
* Comparison of training efficiency between Pandas-based and Dask-based implementations.

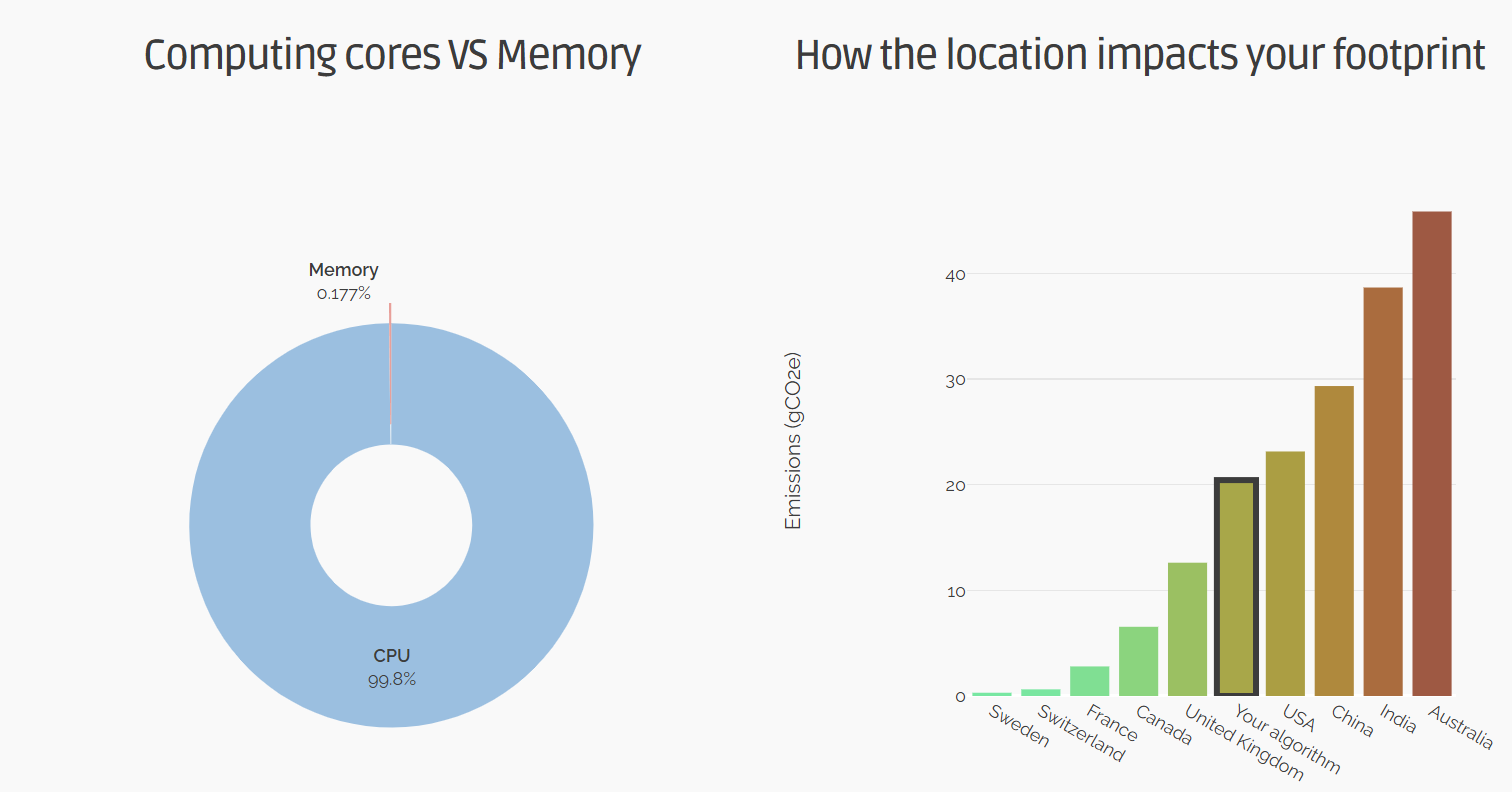
#### ****Dataset Size:****

* Dataset (after preprocessing) contains **79 samples** with **29 features**.
* Experiment evaluates performance on small (Pandas-friendly) dataset.

#### ****Training Setup:****

* **Compute Type:** Local machine.
* **Hardware Used:**
  + CPU: AMD Ryzen 7 5800H
  + RAM: 16GB.
* **Frameworks:** Pandas, Dask, XGBoost.
* **Comparison Metrics:** Training time, memory consumption, energy usage.

In the current implementation, the Green Algorithms calculator only provides approximate estimates due to the short runtime. Nevertheless, the tool offers valuable insights into the relative contributions of computational resources to the overall energy consumption. According to the calculator, in this setup, the CPU accounts for approximately 99,8% of the total energy usage. At the same time, the impact of memory remains insignificant unless workloads involve hundreds of gigabytes of RAM. Additionally, the calculator allows for comparative assessments of how geographic location and the associated energy grid carbon intensity influence the total carbon footprint.



*Figure 1. Green Algorithms calculator report*

### Let’s use the CodeCarbon module to calculate emissions instead of the Green Algorithms calculator.

### 

*Figure 2. Log data about emissions (with CodeCarbon)*

### ****Environmental Impact Assessment****

**Tool Used:** CodeCarbon

* **Pandas Implementation:**
  + Execution time: ≈1.7 seconds
  + Emissions: ≈0.004 gCO₂e
* **Dask Implementation:**
  + Execution time: ≈14.4 seconds
  + Emissions: ≈0.035 gCO₂e

**Insights:**

* Despite Dask being optimized for larger datasets, its use on small data leads to inefficient resource usage and significantly higher emissions.
* CPU usage is the dominant factor (>99%) in both setups due to small data size.
* According to the Green Algorithms calculator, emissions could vary by region based on the carbon intensity of local electricity.

### ****Optimizations for Sustainability****

1. **Choose the Right Tool for the Job:**
   * Use Pandas for small to medium datasets.
   * Use Dask only when data exceeds RAM or needs distributed computing.
2. **Minimize Model Complexity:**
   * Limit the number of boosting rounds.
   * Use early stopping and hyperparameter tuning to avoid overfitting and reduce training time.
3. **Code Efficiency:**
   * Vectorized operations reduce CPU cycles and RAM overhead.
   * Avoid unnecessary/intermediate storage in pipelines.
4. **Location Awareness:**
   * Run training in data centers or regions with low carbon intensity (see fig.1).